Modelling the Behavior of Solaris Resource Manager™ Software

By Engineering Enterprise

Sun BluePrints™ OnLine - August 1999

http://www.sun.com/blueprints

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303 USA
650 960-1300 fax 650 969-9131

Part No.: 806-4637-10
Revision 01, August 1999

Modelling the Behavior of Solaris Resource Manager™ Software

Solaris Research Manager™ (SRM) software lets a system administrator control the distribution of CPU resources to applications. The main benefit of this is that it allows systems to run at much higher utilization levels, with critical applications getting good response times, and less important applications taking the hit when the system gets even busier. To quantify the behavior of SRM software in a CPU bound situation, a model has been produced that predicts the response time of each application as shares of the system are manipulated. The initial model covers a CPU intensive batch environment. In addition, any tendency toward greedy resource consumption by applications can present severe problems for system administration and capacity planning. It is difficult for a human administrator to hover over a server or network and respond to selfish resource consumption. The only way around this kind of threat is adaptive dynamic control of system resources. SRM software achieves this dynamic resource consumption by using a fair share CPU scheduling algorithm.

Performance Guidelines

In the following examples, we modeled batch-type workloads, considering `cpu.shares` only with all relative service demands = 1 and no thinktime, in other words, every job is demanding access to the CPU all the time. This represents a worst-case scenario for expected SRM software performance. Your performance will vary due to either the ameliorating influences of:

- your service demands being less ($D < 1$) relative to our case studies
- your workload intensity being less (think time $Z > 0$) relative to our case studies

or the exacerbating influences of:

- more than one process per-user ($N > 1$) relative to our case studies

- greater disparity in the type of work per group

Top-down Allocations

In general, the allocation of shares should be done top-down starting with the most important groups, users or known service level targets. They need to be guaranteed a minimum response time. To achieve these performance requirements, follow these steps for allocating shares to SRM software groups and users:

- Measure their maximum CPU utilization (Umax) under standard Solaris timeshare software
- Note that the response time (R) is inversely related to Umax
- Allocate CPU shares so that their entitlement ratio is at least equal to Umax
- Use the command `limadm set cpu.shares = $` to set those shares. If these allocations can't be met on a server, go to domains, or split groups across separate servers

SRM Software Capacity Concepts

Although the set it and forget it approach of SRM software is desirable for server capacity administration, dynamically changing SRM software share allocations can have quite unintuitive consequences for user application performance. A system administrator must understand the performance implications of fair share scheduling to make a judicious choice of SRM software allocations.

What is Fair?

In the context of scheduling computer resources and routing packets, the term *fair* is meant to imply equity (not equality) in resource consumption.

The analogy used for the fair share algorithm is financial equity in a corporation. Executive management holds more equity (more financial shares) in a company than underlings. Accordingly, the larger shareholders are entitled to a greater percentage of company profits. Similarly, under SRM software, the more equity a user or group of users holds (via allocated shares), the greater the percentage of server resources they are guaranteed at runtime. Under SRM software there are, however, some significant differences from the way corporate shares work. Three differences that are not immediately apparent are:

- SRM software helps ensure a minimum fraction of CPU capacity, rather than a fixed percentage.
- Allocations of CPU capacity can change dynamically.

- Reallocation is not immediately reflected in CPU capacity.

Here, the minimum fraction is determined from the ratio of a user's shares to the total shares allocated, often referred to as a user's entitlement, as shown in *Equation(1)*. The CPU percentage a user is entitled to (whether the guaranteed minimum or more) is relative, in that it depends on aggregate user activity on the server at any time. As user activity changes, entitlement ratios change, and each user's CPU percentage is modified accordingly. The new entitlements, however, are not immediately reflected in new CPU percentages. This is because the instantaneous CPU usage is only known to the SRM software scheduler by sampling per-process usage at a suitably high rate. All advancement or retardation in the frequency with which a process visits the CPU is handled in SRM software by adjusting process priorities within the CPU run-queue. In some sense, a process is placed incrementally further back or further ahead in the run-queue relative to other processes. Forward migration takes time, so new CPU usage ratios cannot instantly match new entitlements.

Shares, Entitlements and Goals

We can now formalize the previous statements using the following notation:

- $n denotes the shares awarded to user n = 1, 2, 3,....
- En denotes the entitlement of user n.
- Un is the partial CPU utilization of user n.

By entitlement we mean the ratio of a particular user's shares to the total shares allocated for CPU resources. On a server with two users having respective allocations $1 = 1 and $2 = 2, the entitlement for user 1 is given by:

$$E_1 = \frac{\$_1}{\$_1 + \$_2} \qquad -(1)$$

which evaluates to 1/3 or 33% and using the same formula for user 2, we find E2 = 67%.

Next, consider the CPU usage as measured by the SRM software scheduler during an arbitrary period T. This is identical to the CPU busy time (B) of each user process. In *Equation (2)*, with only one user active (E1 = 1.0), the sampled busy time B1 is the same as T for a CPU-bound process i.e., B1 = T. Therefore, the ratio B1/T = 1 which is equivalent to the 100% CPU utilization (U1) seen by user-1. When the second user becomes active, the ratio of entitlements becomes E1/E2 = 0.50 and the SRM software scheduler will adjust each user's CPU usage so that eventually the ratio of partial utilizations reflects that ratio of entitlements, that is U1/U2 = 50%. In general we can say:

$$\lim_{T \to \infty} \frac{U_1}{U_2} \sim \frac{E_1}{E_2} \qquad \text{-(2)}$$

Equation (2) is a formal statement of the goal (or constraint) that the SRM software scheduler aims to achieve. Since entitlement ratios can change dynamically (due to changing user activity), SRM software adjusts the partial CPU utilizations to match them. The mathematical limit on the left side means the utilization ratios can only be achieved in the long run, rather than instantly. Exactly how SRM software accomplishes this matching is described in a previous article from the April 1999 edition of Sun BluePrints™ Online titled, *Solaris Resource Manager™ - Decay Factors And Parameters* (`http://www.sun.com/software/solutions/blueprints/0499/solaris2.pdf`) on SRM Tunables.

Performance Scenarios

The canonical approach to understanding the performance impact of a workload is to start with zero or a small number of users and increase their number incrementally. In the following scenarios, we start with two small-share users and add users and groups of users. Rather than to generate all the possible loadings created by adding incremental users, we will examine four pertinent scenarios.

We consider a server that supports three groups of users: a financial database group (FIN), a web server (WEB), and an operations group (OPS) responsible for support. We treat the FIN and WEB users in aggregate and look at their impact on three explicit users (A, B, C) in detail, starting with just two users (A and B).

Why Use a Performance Model?

Because of the dynamics involved, we chose to incorporate our understanding of SRM software into a performance model using the PDQ Performance Analyzer (`http://members.aol.com/CoDynamo/Prod.PDQ.htm`), which is compatible with the SE performance Toolkit (`http://www.sun.com/sun-on-net/performance/se3/`). Although somewhat unrealistic from a general point of view, these modeling results present insights into potential performance impacts that would be difficult to glean in detailed measurements on a physical server. Eventually, we also want to make this modeling approach available via a web-applet interface to assist you choose beneficial share allocations.

We assume that the CPU service demands (Dcpu) for all users are all equal = 1. To keep our scenarios suitable for straightforward comparisons, all processes are considered to be CPU-bound (worst case batch demand, that is thinktime = 0) and only a single process belongs to each user. The reports generated by the PDQ performance model presented here and are based on the following distribution of 100 shares on the server. Share allocations, entitlements, and the active user shares are shown at the beginning of each table in the following format:

Maximum Share Allocations for Groups and Users

```
Allocations
-----------
100 ACTIVE group cpu.shares out of 100 Allocated.
100 ACTIVE user cpu.shares out of 100 Active group shares.

FIN Group cpu.shares: 60
WEB Group cpu.shares: 10
OPS Group cpu.shares: 30
OPS cpu.shares owned by usrA: 6
OPS cpu.shares owned by usrB: 5
OPS cpu.shares owned by usrC: 19

Group Entitlements
------------------
Group %Active %UserA %UserB %UserC
FIN 60.00 60.00 0.00 0.00
WEB 10.00 10.00 0.00 0.00
OPS 30.00 6.00 5.00 19.00
```

Under SRM, CPU fractions are proportional to the allocated shares for groups and users within a group. These choices are the least upper bounds and they set the stage for using our PDQ/SRM software model to estimate the performance impact of dynamically reassigning CPU capacity due to a change in the number of active shares.

The key performance metric that we (and most SRM software users) are interested in, is response time. Response time is the appropriate performance metric for interactive users and transaction-oriented users that have service level targets. However, here we are looking at a CPU intensive batch workload case, and the response time is interpreted as the time it takes to obtain one second of CPU time for the batch job, with a corresponding reduction in throughput for that batch job. In these cases, we are oversubscribing the CPU by trying to run many more jobs than its capacity and controlling the rate at which jobs complete by varying the shares.

Performance Scenario I - Two Small-Share Users Active

We start with just two users (A and B) active in the OPS group with their respective share allocations $A = 6 and $B = 5. Everyone else is offline. Report I shows that the OPS group gets 100% of the CPU split in the ratio 6/11 for user opsA and 5/11 for user opsB.

PDQ REPORT I

```
Allocations
-----------
11 ACTIVE group cpu.shares out of 100 Allocated.
11 ACTIVE user cpu.shares out of 11 Active group shares.

FIN Group cpu.shares: 60 (offline)
WEB Group cpu.shares: 10 (offline)
OPS Group cpu.shares: 30
OPS cpu.shares owned by usrA: 6
OPS cpu.shares owned by usrB: 5
OPS cpu.shares owned by usrC: 19 (offline)

Group Entitlements
------------------
Group %Active %UserA %UserB %UserC
FIN 0.00 0.00 0.00 0.00
WEB 0.00 0.00 0.00 0.00
OPS 100.00 54.55 45.45 0.00

User Workload Parameters
------------------------
User Procs Think Dcpu
opsA 1.00 0.00 1.0000
opsB 1.00 0.00 1.0000

Estimated SRM Performance
-------------------------
User Thru RTime %Ucpu
opsA 0.56 1.80 55.51
opsB 0.44 2.25 44.49

Comparative Timeshare Performance
--------------------------
User Thru RTime %Ucpu
opsA 0.50 2.00 50.00
opsB 0.50 2.00 50.00
```

The CPU fractions, corresponding to share allocations, are reflected in the %Ucpu field of the Estimated SRM Performance section of *PDQ REPORT I*. Many different comparisons can be made using all this data. For brevity, we consider only the following three:

- Comparing SRM to timeshare response times within a scenario
- Comparing SRM response times between two successive scenarios
- Comparing SRM response times between any two scenarios

The following table (TABLE 1) will help us make those comparisons.

TABLE 1 **Comparison of Response Times in Report I**

User	Rsrm	Rts	Rsrm / Rts
opsA	1.80	2.00	0.90
opsB	2.25	2.00	1.13

The column headings have the following meaning: `Rsrm` is the response time under the SRM software scheduler, `Rts` is the corresponding response time under the conventional Solaris timeshare scheduler. The last column shows the ratio of these two times. If `Rsrm/Rts` < 1, then SRM software can be expected to give better performance than timeshare. Otherwise, `Rsrm/Rts` > 1, and SRM software can be expected to give worse performance than timeshare under the selected share allocations.

In Scenario I (TABLE 1), because user opsA has 10% more entitlement than they would under a timeshare scheduler, their response time is better (`Rsrm/Rts` < 1) than it would be under timeshare by 10%, when running CPU-bound. On the other hand, user opsB (having only one less share) suffers more than a 10% degradation in response time relative to that expected under timeshare scheduler.

Performance Scenario II - Three Users Active

Now, we consider the impact on response times of bringing user opsC online within the OPS group. The results are shown in the following report:

PDQ REPORT II

```
Allocations
-----------
30 ACTIVE group cpu.shares out of 100 Allocated.
30 ACTIVE user cpu.shares out of 30 Active group shares.

FIN Group cpu.shares: 60 (offline)
WEB Group cpu.shares: 10 (offline)
OPS Group cpu.shares: 30
OPS cpu.shares owned by usrA: 6
OPS cpu.shares owned by usrB: 5
OPS cpu.shares owned by usrC: 19

Group Entitlements
------------------
Group %Active %UserA %UserB %UserC
FIN 0.00 0.00 0.00 0.00
WEB 0.00 0.00 0.00 0.00
OPS 100.00 20.00 16.67 63.33

User Workload Parameters
------------------------
User Procs Think Dcpu
opsA 1.00 0.00 1.0000
opsB 1.00 0.00 1.0000
opsC 1.00 0.00 1.0000


Estimated SRM Performance
-------------------------
User Thru RTime %Ucpu
opsA 0.19 5.34 18.72
opsB 0.17 5.80 17.25
opsC 0.64 1.56 64.02


Comparative Timeshare Performance
--------------------------
User Thru RTime %Ucpu
opsA 0.33 3.00 33.33
opsB 0.33 3.00 33.33
opsC 0.33 3.00 33.33
```

In addition to the columns in the *Comparison of Response Times in Report I* (TABLE 1), we added a new column to compare the performance ratio between this Scenario II and Scenario I. `Rsrm2` is the response time in Scenario II and `Rsrm1` is the response time in Scenario I (see TABLE 2).

TABLE 2 **Comparison of Response Times in Report II**

User	Rsrm	Rts	Rsmr2 / Rsrm1
opsA	5.34	3.00	2.97
opsB	5.80	3.00	2.58
opsC	1.56	3.00	N / A

Because user opsC is entitled to 63% of the CPU, their response time is not only better than the other two OPS users, but twice as good as it would be under timeshare (`Rsrm/Rts` = 0.52). However, opsA's response time is now almost three times worse (`Rsrm2/Rsrm1` = 2.97) than it was when opsC was absent, and user opsB's performance is about 2.5 times worse (`Rsrm2/Rsrm1` = 2.58) than it was in opsC's absence.

Moreover, when user opsC comes online, they are entitled to 63% of the capacity so, they will expect their response time to be better than it would be under conventional Solaris timeshare scheduler.

Performance Scenario III - Two Groups Active

Next, we consider the impact on response times of bringing WEB users online. *PDQ Report III* summarizes the expected performance impact.

PDQ REPORT III

```
Allocations
-----------
40 ACTIVE group cpu.shares out of 100 Allocated.
40 ACTIVE user cpu.shares out of 40 Active group shares.

FIN Group cpu.shares: 60 (offline)
WEB Group cpu.shares: 10
OPS Group cpu.shares: 30
OPS cpu.shares owned by usrA: 6
OPS cpu.shares owned by usrB: 5
OPS cpu.shares owned by usrC: 19

Group Entitlements
------------------
Group %Active %UserA %UserB %UserC
FIN 0.00 0.00 0.00 0.00
WEB 25.00 25.00 0.00 0.00
OPS 75.00 15.00 12.50 47.50

User Workload Parameters
------------------------
User Procs Think Dcpu
wAgg 1.00 0.00 1.0000
opsA 1.00 0.00 1.0000
opsB 1.00 0.00 1.0000
opsC 1.00 0.00 1.0000


Estimated SRM Performance
-------------------------
User Thru RTime %Ucpu
wAgg 0.25 4.00 25.00
opsA 0.14 6.97 14.35
opsB 0.13 7.88 12.69
opsC 0.48 2.08 47.96


Comparative Timeshare Performance
---------------------------------
User Thru RTime %Ucpu
Web 0.25 4.00 25.00
opsA 0.25 4.00 25.00
opsB 0.25 4.00 25.00
opsC 0.25 4.00 25.00
```

The response time results from Scenarios I, II, and III are shown in the following table (TABLE 3):

TABLE 3 **Comparison of Response Times in Report III**

User	Rsrm	Rts	Rsrm / Rts	Rsrm3 / Rsrm2
wAgg	4.00	4.00	1.00	N / A
opsA	6.97	4.00	1.74	1.31
opsB	7.88	4.00	1.97	1.36
opsC	2.08	4.00	0.52	1.33

We can draw the following striking conclusions. In the presence of the WEB group, all the OPS users suffer a 31% to 36% degradation relative to their SRM response times in Scenario II (TABLE 2).

In particular, user opsC suffers a 33% degradation when WEB users becomes active even though they have the greatest CPU entitlement within the OPS group. On the other hand, opsC is still doing almost twice as well (`Rsrm/Rts` = 0.52) as they would be doing under a timeshare scheduler.

Performance Scenario IV - All Groups Active

Finally, we consider the impact on response times of bringing the FIN group online. *PDQ Report IV* summarizes the expected performance impact.

PDQ REPORT IV

```
Allocations
-----------
100 ACTIVE group cpu.shares out of 100 Allocated.
100 ACTIVE user cpu.shares out of 100 Active group shares.

FIN Group cpu.shares: 60
WEB Group cpu.shares: 10
OPS Group cpu.shares: 30
OPS cpu.shares owned by usrA: 6
OPS cpu.shares owned by usrB: 5
OPS cpu.shares owned by usrC: 19
```

PDQ REPORT IV (Cont'd)

```
Group Entitlements
------------------
Group %Active %UserA %UserB %UserC
FIN 60.00 60.00 0.00 0.00
WEB 10.00 10.00 0.00 0.00
OPS 30.00 6.00 5.00 19.00

User Workload Parameters
------------------------
User Procs Think Dcpu
fAgg 1.00 0.00 1.0000
wAgg 1.00 0.00 1.0000
opsA 1.00 0.00 1.0000
opsB 1.00 0.00 1.0000
opsC 1.00 0.00 1.0000


Estimated SRM Performance
-------------------------
User Thru RTime %Ucpu
fAgg 0.60 1.67 60.00
wAgg 0.10 10.00 10.00
opsA 0.02 55.62 5.99
opsB 0.01 66.74 4.99
opsC 0.06 17.60 18.94


Comparative Timeshare Performance
--------------------------
User Thru RTime %Ucpu
Fin 0.20 5.00 20.00
Web 0.20 5.00 20.00
opsA 0.20 5.00 20.00
opsB 0.20 5.00 20.00
opsC 0.20 5.00 20.00
```

Because FIN owns the majority of the total shares, the performance impact on other users is quite significant.

TABLE 4 **Comparison of Response Times in Report IV**

User	Rsrm	Rts	Rsrm / Rts	Rsrm4/Rsrm3
fAgg	1.67	5.00	N / A	N / A
wAgg	10.00	5.00	2.00	2.50
opsA	55.62	5.00	11.12	7.98
opsB	66.74	5.00	13.35	8.47
opsC	17.60	5.00	3.52	8.46

OPS user's response time now suffers a degradation factor of about *8x* relative to FIN being offline in Scenario III (see TABLE 4). But from the viewpoint of our original pair of OPS users (opsA and opsB) in Scenario (TABLE 1), the news is much worse. They see about a 30x degradation relative to the original configuration in *PDQ Report I*. Note also in the 4th column that opsA and opsB are doing worse by more than 10x relative to timeshare scheduling.

- Allowing a sudden large increase in response times by an order of magnitude or more is undesirable and may be something you can avoid by allocating SRM software shares in a more balanced way.

This is a worst case scenario in that we have been looking at the impact of increasingly heavy-share users on the smaller-share users (especially opsA and opsB). We can turn the question around and ask what is the impact on the heavy-share users of bringing a lighter weight (but not insignificant) user online, for example opsC. Without showing all the details, the following table (TABLE 5) summarizes the performance data with opsC offline.

TABLE 5 **Comparison of Response Times with opsC Offline**

User	Rsrm	Rts	Rsrm / Rts	Rsrm / Rsrm4
fAgg	1.35	4.00	0.34	0.81
wAgg	8.10	4.00	2.03	0.81
opsA	99.42	4.00	24.86	1.79
opsB	119.52	4.00	29.88	1.79

Even looking at this from this alternative perspective of a single user (opsC) impacting a group response time like FIN (with 60% of the shares), we see that WEB and FIN response times can be impacted by as much as 20% degradation.

- You need to be fully aware of any business service-level agreements that pertain to groups when allocating shares.

Although a more benign effect, the impact of an SRM software user on a different SRM software group can still be very significant when you take service targets into account.

Metrics to Monitor

At the moment, none of the generic Solaris Operating Environment software performance tools provide the exact performance data that is needed to easily monitor SRM software technology-based performance. The closest you can come at present is to monitor and log per-process CPU consumption with a command like `/usr/ucb/ps aux` or `pw.se` from the SE performance Toolkit (`http://www.sun.com/sun-on-net/performance/se3/`). The `pw.se` command can be setup to pattern match on the users in each share group so that the resource consumption on a per share basis is visible. Recalling that the goal of SRM software is expressed in *Equation (2)* in the earlier discussion, the %CPU and TIME fields in the ps command can be used to give some idea of how well SRM software is achieving the goal set for it under a top-down allocation of shares.